Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently amended) A computer program product for performing format conversions, tangibly stored on a computer-readable medium, comprising instructions operable to cause a programmable processor to:

identify an ~~first representation of a persistent~~ object and a first format indicator associated with the ~~persistent~~ object;

identify a ~~current~~ second format indicator associated with a secondary program called by a primary program; and

apply a conversion engine, wherein the conversion engine takes as inputs

i) a first schema referenced by the first format indicator and defining a data format of the object, the first schema being part of the secondary program,

ii) a second schema referenced by the ~~current~~ second format indicator and defining a current data format of the secondary program, the second schema being part of the secondary program, and

iii) the ~~first representation of the persistent~~ object, ~~and~~

wherein the conversion engine ~~generates a second representation in a format referenced by the current format indicator~~ reformats the object so that it conforms to the current data format of the secondary program. ~~; and~~

~~read the second representation to recreate the persistent object.~~

2. (Original) The computer program product of claim 1, wherein the format indicators are format numbers.

3. (Cancelled)

4. (Currently amended) The computer program product of claim [[3]]1, wherein the secondary program is a plug-in and the primary program is an application program associated with the plug-in.

5. (Currently amended) The computer program product of claim 1, wherein the conversion engine cooperates with a ~~second~~ auxiliary conversion engine capable of converting formats without reference to the schema.

6. (Currently amended) The computer program product of claim 5, wherein the conversion engine and auxiliary conversion engine are part of a single conversion service contained in a single application program.

7. (Currently amended) A file conversion system, comprising:

a primary program;

a secondary program called by the primary program;

at least one ~~file associated with the primary program, the at least one file including an first representation of a persistent~~ object created by the secondary program;

a conversion service contained in the primary program;

a set of schemas contained in the secondary program, the schemas describing the format in which previous versions of the secondary program wrote data ~~into the at least one file~~ and the format in which the current version of the secondary program writes data;

wherein the conversion service is capable of converting the ~~first representation of a persistent~~ object from a first format to a second format based on the format information provided by the set of schemas.

8. (Original) The system of claim 7, wherein the secondary program is a plug-in.

9. (Currently amended) The system of claim 8, wherein the conversion service cooperates with a ~~second~~ auxiliary conversion service capable of converting data from one format to another without reference to the schema.

10. (Currently amended) The system of claim 9, wherein the conversion service and auxiliary conversion service are part of a ~~single hybrid conversion service contained in~~ a single application program.

11. (Currently amended) A method for performing format conversions, comprising:

finding an ~~first representation of a persistent~~ object and first format indicator associated with the ~~persistent~~ object;

finding a current format indicator associated with a secondary program called by a primary program;

applying a conversion engine, wherein the conversion engine takes as inputs a first schema defining a data format of the object, contained in the secondary program and referenced by the first format indicator, a second schema referenced by the current format indicator, contained in the secondary program and defining a data format associated with the secondary program and the ~~first representation of the persistent~~ object ~~and~~ , wherein the conversion engine ~~generates a second representation in a format referenced by the current format indicator~~ reformats the object so that it conforms to the current data format of the secondary program~~; and~~

~~reading the second representation to recreate the persistent object~~.

12. (Original) The method of claim 11, wherein the format indicators are format numbers.

13. (Original) The method of claim 11, wherein the schema are contained in a secondary program controlled by a primary program.

14. (Original) The method of claim 13, wherein the secondary program is a plug-in and the primary program is an application program associated with the plug-in.

15. (Currently amended) The method of claim 11, wherein the conversion engine cooperates with an ~~second~~ auxiliary conversion engine capable of converting formats without reference to the schema.

16. (Currently amended) The method of claim 15, wherein the conversion engines and auxiliary conversion engine are part of a single conversion service contained in a single application program.

17. (Currently amended) A method for performing format conversions, comprising:

a) access ~~read a file comprising~~ a plurality of ~~persistent~~ objects written by one or more secondary programs called by a primary program;

b) identify a current format indicator associated with a format compatible with a current version of a secondary program;

c) identify a first format indicator associated with a first object created by a version of the secondary program;

d) compare the format indicators to determine whether the ~~persistent~~ first object is in a format compatible within the current version of the secondary program;

e) where the format of the persistent object is not compatible, apply a conversion engine in a primary program to convert the ~~data of the persistent~~ first object to a compatible format based upon format information ~~that is~~ associated with the format indicators and contained within the secondary program;

f) identify another format indicator associated with another object; and

g) repeat steps (d)-(f) to convert the remaining ~~persistent~~ objects to formats compatible with the current versions of the one or more secondary programs.

18. (Currently amended) A method for performing reverse format conversions, comprising:

finding a current format indicator;

querying a conversion service associated with a secondary program to identify a target format indicator associated with a format in which a previous version of the secondary program writes data;

applying a conversion engine in a primary program associated with the secondary program, wherein the conversion engine takes as inputs i) current and target schema referenced by the format indicators and stored within the secondary program, said current schema and target schema defining formats in which the secondary program currently writes and previously wrote data, respectively, and ii) a data ~~first representation of a persistent~~ object,

whereby the conversion engine reformats the object so that it conforms to the target schema ~~generates a second representation of the persistent object in a format referenced by the target format indicator~~.

19. (Cancelled)

20. (Cancelled)

21. (New) The computer program product of claim 4, wherein the conversion service receives as inputs schemas contained in a plurality of different plug-ins and wherein the conversion service reformats objects associated with a plurality of different plug-ins.